



PATENT  
E \$ 2700

In the Application of

Sunshin AN et al.

Application No.: 09/395,207

: Group Art Unit: 2155

Confirm. No.: 5612

: Examiner: Young N. WON

Filed: September 14, 1999

: Customer No.: 34610

For: NETWORK MANAGEMENT SYSTEM AND METHOD

**TRANSMITTAL OF APPEAL BRIEF**

U.S. Patent and Trademark Office  
2011 South Clark Place  
Customer Window, Mail Stop Appeal Brief-Patents  
Crystal Plaza Two, Lobby, Room 1B03  
Arlington, VA 22202

RECEIVED

NOV 17 2003

Technology Center 2100

Sir:

Submitted herewith in triplicate is Appellant(s) Appeal Brief in support of the Notice of Appeal filed May 29, 2003. Enclosed is Check No. 10615 for the Appeal Brief fee of \$330.00.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 16-0607 and please credit any excess fees to such deposit account.

Respectfully submitted,  
FLESHNER & KIM, LLP

John L. Lusk

John C. Eisenhart  
Registration No. 38,128

P.O. Box 221200  
Chantilly, Virginia 20153-1200  
703 502-9440<sub>JCE/jlg</sub>  
**Date: November 12, 2003**

**Please direct all correspondence to Customer Number 34610**

Docket No.: K-105

#17  
LOT  
11-20-03  
PATENT

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE  
BEFORE THE BOARD OF APPEALS AND INTERFERENCES**

In re Application of

Sunshin AN, Dongjin HAN, Wenzhe CUI,  
Youngeun PARK, Shinyuk KANG,  
and Soo Hyun PARK

Serial No.: 09/395,207

Confirm. No.: 5612

Filed: September 14, 1999

For: NETWORK MANAGEMENT SYSTEM AND METHOD



Group Art Unit: 2155

Examiner: Young N. WON

Customer No.: 34610

**APPEAL BRIEF**

**RECEIVED**

NOV 17 2003

Technology Center 2100

U.S. Patent and Trademark Office  
2011 South Clark Place  
Customer Window, Mail Stop Appeal Brief-Patents  
Crystal Plaza Two, Lobby, Room 1B03  
Arlington, VA 22202

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed May 29, 2003.

**I. INTRODUCTION**

This is an appeal from a January 29, 2003 Final Office Action rejecting claims 1-15, 17, and 18.

11/14/2003 AMONDAF1 00000024 09395207

01 FC:1402

330.00 DP

**REAL PARTY IN INTEREST**

The real party in interest for this appeal and the present application is LG Information & Communications, Ltd., by way of an Assignment recorded in the U.S. Patent and Trademark Office at Reel 010471, Frame 0169.

**RELATED APPEALS AND INTERFERENCES**

There are presently no appeals or interferences, known to Appellants, Appellants' representative, or the Assignee which will directly affect or be directly affected by or have a bearing upon the Board's decision in the pending appeal.

**STATUS OF THE CLAIMS**

Claims 1-15, 17, 18 are pending in the application. The application was originally filed with claims 1-18. Claim 16 was canceled by a Reply filed on April 29, 2003.

Independent claim 1 stands finally rejected and is on appeal. In addition, claims 2-7, which depend from independent claim 1, stand finally rejected and are on appeal.

Independent claim 8 stands finally rejected and is on appeal. In addition, claims 9-14, which depend from independent claim 8, stand finally rejected and are on appeal.

Independent claim 15 stands finally rejected and is on appeal. In addition, claims 17 and 18, which depend from independent claim 15, stand finally rejected and are on appeal.

### **STATUS OF AMENDMENTS**

In response to an Office Action dated July 25, 2002, Applicants filed an Amendment on November 25, 2002, amending claims 1 and 8. In response to a Final Office Action dated January 29, 2003, Applicants filed a Reply dated April 29, 2003, amending claims 1, 8, and 15 and canceling claim 16. An Advisory Action was issued on May 12, 2003, which maintained the rejections, but entered the Reply dated April 29, 2003. In response to the Advisory Action, Applicants filed a Notice of Appeal. A copy of all pending claims is attached hereto as Appendix A.

### **SUMMARY OF THE INVENTION**

#### **A. General**

The present invention relates to a network management system capable of updating or adding a managed object without stopping an on-going network management system. Thus, referring to Figure 1 of the preferred embodiment, a kernel 14 preferably runs in the form of a thread and is responsible for initializing a managed system, establishing an association with other management systems, performing management operations and updating new MO information without recompiling or restarting the managed system.

The full set of MO instances available across a management interface is organized into a tree structure in the containment tree 16. When an operation is performed in a managed system, all MO instances are accessed through the containment tree 16. The MOF 18 maintains the information on all MO classes. Utilizing the GDMO compiler 22, the class

information is generated from GDMO scripts and stored in specific files called External Meta file (EMM) 26. The EMM 26 is a special set of files including several files where each file contains information of MO class elements. The output of GDMO compiler 22 includes MO class codes.

The MO class codes are compiled into the dynamic library 24 preferably in a form of a dynamic link library by using an appropriate compiler, and the EMM 26 is used to initialize and update the MOF when the MO classes are added into a network management system. Thus, the MO class information is preferably completely separated from the MO instance implementation. Additionally, the EMM has all managed instances. Thus, no MIB tree needs to be created each time an update is necessary. Hence an update can be performed without interrupting the operation.

## **B. The Claimed Invention**

### **1. Group I - Claims 1-7**

Independent claim 1 recites a network management system, including a management system kernel that provides management systems with a run-time environment, and a managed object generation environment that provides a development environment for managing applications, wherein the management system kernel can at least one of dynamically add and dynamically modify managed object (MO) information based upon an external meta file (EMM) from the managed object generation environment without interrupting an operation

of the network management system.

Additional features of the claimed invention are found in dependent claims 2-7.

Claim 2 depends from claim 1, and recites that the management system kernel includes a communication module that provides communication with a network manager, a managed object framework that maintains information on MO classes, a kernel that stores a dynamic class loading module and initializes the network management system, wherein said kernel establishes an association with other management systems through the communication module, performs management operations on MOs, adds the MO information in the managed object framework using the dynamic class loading module and modifies the MO information in the managed object framework using the dynamic class loading module, and a containment tree that organizes MO instances according to the information on MO classes and allows access to the MO instances when a management operation is performed in the network managed system.

Claim 3 depends from claim 2, and recites that the managed object framework maintains information on MO classes by registering MO class codes on a class information table.

Claim 4 depends from claim 2, and recites that the kernel creates at least one dedicated agent to perform subsequent management operations from management systems with which an association has been established.

Claim 5 depends from claim 1, and recites that the managed object generation

environment includes a MO compiler that compiles a MO script to generate the EMM file and MO class codes, and a dynamic library storing the MO class codes.

Claim 6 depends from claim 5, and recites that the EMM file includes MO class definition described in the MO script, and identifies a location and name in the dynamic library of a corresponding MO class.

Claim 7 depends from claim 5, and recites that the MO class codes are compiled and stored in the dynamic library in a form of a dynamic link library.

## **2. Group II - Claim 8-14**

Independent claim 8 recites a network management method including (a) storing a dynamic class loading routine in a management system kernel, (b) initializing a managed system by constructing a managed object framework of the management system kernel that contains information of managed object (MO) classes, (c) creating MO instances and registering the MO instances in a containment tree of the management system kernel according to the information of MO classes, (d) checking whether a dynamic class loading flag is on when receiving a management operation request from a management system, and (e) updating MO information on the management system kernel without interrupting an operation of the management system by, waiting for all threads to complete execution, loading a dynamic library to the managed object framework utilizing the dynamic class loading routine when the dynamic class loading flag is on, and resetting the dynamic class loading flag to off.

Additional features of the claimed invention are found in dependent claims 9-14.

Claim 9 depends from claim 8, and adds the feature (f) performing the requested management operation and sending a management operation result to the management system requesting the management operation when the dynamic class loading flag is not on.

Claim 10 depends from claim 8, and recites that the dynamic class loading routine of (e) includes opening a EMM file stored outside the management system kernel, and loading the dynamic library indicated by the EMM file.

Claim 11 depends from claim 10, and adds the features of storing information about the management system requesting a management operation.

Claim 12 depends from claim 10, and adds the features of checking whether an additional thread can be created, creating a dedicated agent to take charge of subsequent management operations from the management system requesting an association if an additional thread can be created, and executing the dedicated agent thread and delivering association and management operation information to the dedicated agent to be utilized in interacting with the management system.

Claim 13 depends from claim 8, and recites that the dynamic class loading flag is set on when the management system requesting a management operation invokes the dynamic class loading function to perform one of adding and modifying MO information in the management system kernel.



Claim 14 depends from claim 13, and recites that the management system requesting the management operation invokes the dynamic class loading function by sending a control signal.

**3. Group III - Claim 15, 17, and 18**

Independent claim 15 recites a network management method, including storing a dynamic class loading routine in a management system kernel of the managed system, updating the management system kernel by modifying managed object (MO) information in the management system kernel while the managed system is operating by utilizing the dynamic class loading routine, and generating the MO information to be modified and generating a external meta file (EMM) in a managed object generation environment of the managed system wherein the dynamic class loading routine opens the EMM file to modify the MO information in the management system kernel.

Additional features of the claimed invention are found in dependent claims 17 and 18.

Claim 17 depends from claim 16, and recites that the MO information to be modified is stored in the managed object generation environment in the form of a dynamic link library.

Claim 18 depends from claim 17, and recites that the EMM indicates an address of a dynamic link library corresponding to the MO information to be modified, and wherein the MO information is modified in the management system kernel according to said address.

### **III. THE ISSUES AND REJECTIONS**

Claims 1-3, 5-11, 13-15, 17, and 18 stand rejected under 35 U.S.C. § 103(a) over Bentley et al. (U.S. Patent No. 5,815,415) (hereinafter Bentley) in view of Lin et al. (U.S. Patent No. 6,243,457 B1) (hereinafter Lin). Claim 4 stands rejected under 35 U.S.C. § 103(a) over Bentley in view Applicant's background art. Claim 12 stands rejected under 35 U.S.C. §103(a) over Bentley, in view of Applicant's background art, and further in view of Sheard et al. (U.S. Patent No. 6,208,345) (hereinafter Sheard).

#### **A. Issues**

1. Does Bentley, in view of Lin, render obvious the subject matter of claims 1-3, 5-11, 13-15, 17, and 18 under 35 U.S.C. § 103(a);
2. Does Bentley, in view of Applicant's background art, render obvious the subject matter of claim 4 under 35 U.S.C. § 103(a);
3. Does Bentley, in view of view of Applicant's background art, and further in view of Sheard render obvious the subject matter of claim 12 under 35 U.S.C. § 103(a).

### **IV. GROUPING OF CLAIMS**

Claims 1-7 comprise Group 1, wherein claims 1-3 and 5-7 stand or fall together, and claim 4 stands or falls alone.

Claims 8-14 comprise Group 2, wherein claims 8-11, 13, and 14 stand or fall together, and claim 12 stands or falls alone..

Claims 15, 17, and 18 comprise Group 3, wherein claims 15, 17, and 18 stand or fall together.

## **V. ARGUMENT**

As demonstrated below, the rejections are based upon imputing to the references that which they do not teach or suggest. Such is not a proper rejection.

### **A. Scope And Content Of The Prior Art**

#### **1. Bentley (U.S. Patent No. 5,815,415)**

Bentley relates to a computer aided design system, generally referred to as a computerized modeling system (CMS). Bentley teaches that the architecture of the CMS includes various layers, including a static kernel 12, a dynamic framework 14, and a portable persistent model 16. The kernel 12 provides the services necessary to load and execute the higher levels.

Referring to column 8, lines 4-7, Bentley teaches that the kernel 12 can be extended by loading additional dynamic modules 23 with associated DLS files. The DLS files are Dynamic Link Specification files, and are provided within the kernel 12. Accordingly, they are not external to the kernel 12. Furthermore, they are not external meta files. The DLS files are simply used to resolve the addresses of the native code functions in the kernel 12, not a dynamic library loaded to the managed object framework.

Bentley further teaches that the kernel 12 includes an object manager 30 that provides for the allocation, references, and persistence of all objects in the portable persistent model 16. Thus, whenever a new object is created, the object/persistence manager 30 creates an object descriptor for the object. See column 8, lines 18-28.

There is no disclosure in Bentley, however, of dynamically modifying managed object information based upon an external meta file from the managed object generation environment without interrupting an operation of the network management system.

Although at column 16, lines 47-55, Bentley teaches a "meta-class," the Bentley meta-class merely contains a description of an associated class object. Additionally, even if the meta-class were equivalent to the claim 1 meta file there is no teaching that the meta-class is external and used by the kernel to dynamically add and dynamically modify managed object information without interrupting an operation of the network management system.

In addition, Bentley discloses having a dynamic method dispatching. It is constructed at run-time, however, Bentley suggests that only the method's names are added and compiled at a run-time, not the entire program. Therefore, Bentley merely includes a system, which dynamically add and compile the method's names as well as the entire program at run-time. See column 17, lines 44-67 and column 18, lines 1-23.

**2. Lin et al. (U.S. Patent No. 6,243,457 B1)**

Lin relates to an Intelligent Network wherein a service application process may be deployed without restricting user access to the Intelligent Network during adding and updating information. In particular, Lin teaches using separate MIB trees for service applications and the service control point platform MIB tree. Referring to Lin, column 7, lines 58-66, Lin requires the use of separate MIB trees for each update and each new service addition. Once the information is loaded, the original MIB tree is deleted. See Lin, Column 9, lines 55-65. Thus, one containment tree comprises the Service Control Point platform function while a separate containment tree is adopted for each new service application. See Lin, Column 4, lines 13-17. Accordingly, Lin does not teach an external meta file used by the kernel to dynamically add and dynamically modify managed object information without interrupting an operation of the network management system. Furthermore, Lin fails to teach loading a dynamic library to the managed object framework utilizing the dynamic class loading routine when the dynamic class loading flag is on, and resetting the dynamic class loading flag to off.

**3. Sheard et al. (U.S. Patent No. 6,208,345)**

Sheard relates to a visual data integration system architecture and method. In particular, Sheard teaches a single user interface that provides information to users. Sheard also teaches that a processing thread may make use of additional system resources. See column 15, lines 47-50. Hence, Sheard teaches a technology-independent integration mechanism that provides

for the exchange of technology-dependent data between disparate applications. Sheard further teaches that the integrated information system is developed visually by using a visual interface to drag-and-drop components within an area of the interface. Moreover, each of the components is taught to be a graphical representation of various telecommunications hardware and software elements, such as information stores, processors, and input/output devices. The graphical interconnections are then converted into logical or physical interconnections by an underlying configuration framework operating above and in concert with the transport framework. Format neutral data meta-models are only used to model the input and output data requirements of disparate systems and system components. This is done to remove any cross-dependencies that exist between the systems and technologies implicated in a data integration project.

Sheard, however, fails to teach or suggest creating a dedicated agent that takes charge of subsequent management operations from the management system requesting an association if an additional thread can be created.

#### 4. Applicant's Background Art

Applicant's Background Art teaches an OSI system management structure, where a system manager and an agent model are utilized such that the agent can manage objects. A managed object (MO) is generally an OSI abstract view of a logical or physical system resource to be managed. A MO class defines attributes of an object and a particular set of attributes is

a MO instance. The MO instances have many relationships, of which the containment relationship is the primary relationship. Accordingly, the agent maintains a containment tree to express the containment relationships of the MO class instances.

The containment tree is a starting point of the agent's management operation as well as the core of MO management. The agent executes scoping of a priority containment tree in response to a management request from the manager. The scoping performed by the agent entails selecting scopes of the MO instances to be managed and carrying out the necessary management operation on the selected MO instance. The agent may also perform filtering by which a determination is made whether the management operations should be performed on the scoped MO.

Moreover, Applicant's Background Art teaches that to add a new MO class or to modify a MO class in the network management system, an on-going network management system must be stopped before a MO can be added or modified, and then re-executed after the addition or modification. Additionally, the process of stopping in the middle of operation and re-execution may take hours or even days depending upon the addition or modification. Hence, as numerous users utilize a network management system, stopping the network management system for even a short period of time is a fatal defect or a significant disadvantage to the network management system and users, and thereby the communication service.

**B. Lack of Motivation to Combine**

Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. In re Fine, 837 F.2d 1071, 5 U.S.P.Q.2d (BNA) 1596 (Fed. Cir. 1988); In re Jones, 958 F.2d 347, 21 U.S.P.Q.2d (BNA) 1941 (Fed. Cir. 1992). Additionally, it is improper to combine references where the references teach away from their combination. In re Grasselli, 713 F.2d 731, 743, 218 USPQ 769, 779 (Fed. Cir. 1983).

It is respectfully submitted that one of skill in the art would not have been motivated to selectively combine features of Bentley and Lin to produce the claimed devices and methods. Specifically, it is respectfully submitted that one of skill in the art would not have been motivated to modify the Bentley system management technique, which is designed for one type of management, such that it incorporates a different style of system management, as taught by Lin. Specifically, the two references handle managed objects in distinctly different ways, and it would not have been obvious to selectively combine features of these disparate systems to yield the claimed system and method.

Moreover, there is no teaching or suggestion in the references for such a combination. For example, the Patent Office asserts that would have been obvious to combine the teachings of Bentley and Lin, because "implementing modifications of the managed object information without interrupting an operation of the network management system... would save time by



eliminating the need for re-initialization or rebooting of all devices." See Office Action dated January 29, 2003, Page 6. Although, as taught by Applicant's present disclosure, such a combination is beneficial, there is no teaching or suggestion within the references for such a combination. Moreover, because of the benefit of efficiency, if the proposed combination were obvious, Applicant submits that it would have been discussed in the references.

For all the reasons given above, it is respectfully submitted that it requires impermissible hindsight reconstruction, in view of Applicants' own invention, to selectively combine features of the references applied in the Office Action to achieve the claimed devices and methods. For at least these reasons, it is respectfully submitted that all of the Section 103 claim rejections are improper and should be withdrawn.

**D. Additional Arguments Regarding Each Issue**

**1. Issue 1: Claims 1-3, 5-11, 13-15, 17, 18**

Claims 1-3, 5-11, 13-15, 17, and 18 stand rejected under 35 U.S.C. § 103(a) over Bentley et al. (U.S. Patent No. 5,815,415) (hereinafter Bentley) in view of Lin et al. (U.S. Patent No. 6,243,457 B1) (hereinafter Lin). Because the asserted references fail to disclose all the claimed features, it is respectfully submitted that the rejection is improper and should be withdrawn.

The asserted combination of references fails to teach or suggest at least a management system kernel that provides management systems with a run-time environment, and a managed

object generation environment that provides a development environment for managing applications, wherein the management system kernel can at least one of dynamically add and dynamically modify managed object information based upon an external meta file from the managed object generation environment without interrupting an operation of the network management system, as recited in claim 1.

Additionally, the asserted combination fails to teach or suggest at least a network management method including, inter alia, storing a dynamic class loading routine in a management system kernel, and updating managed object information on the management system kernel without interrupting an operation of the management system by ... loading a dynamic library to the managed object framework utilizing the dynamic class loading routine, as recited in claim 8.

Finally, the asserted combination fails to teach or suggest at least a network management method including, inter alia, storing a dynamic class loading routine in a management system kernel of the managed system, and updating the management system kernel by modifying managed object information in the management system kernel while a managed system is operating by utilizing the dynamic class loading routine, and generating the MO information to be modified and generating an external meta file (EMM) in a managed object generation environment of the managed system wherein the dynamic class loading routine opens the EMM file to modify the MO information in the management system kernel, as recited in claim 15.

For at least above discussed reasons, the asserted combination of references fails to teach or suggest all of the claimed features, as required by Section 103. Because a prima facie case of obviousness cannot be made, it is respectfully submitted that the rejection is improper and should be withdrawn.

Claims 2, 3, and 5-7 depend from claim 1, claims 9-11, 13, and 14 depend from claim 8, and claim 17 and 18 depend from claim 15. These claims are allowable for at least the same reasons discussed above with respect to the corresponding independent claims. Consequently, it is respectfully submitted that the rejection of these claims is also improper, and should likewise be withdrawn.

**2. Issue 2: Claim 4**

Claim 4 stands rejected under 35 U.S.C. § 103(a) over Bentley in view Applicant's Background Art.

The asserted combination of references fails to establish a prima facie case of obviousness, as required by Section 103. For example, claim 4 depends from claim 1. As discussed above, Bentley fails to teach or suggest all the features of claim 1. Moreover, the recited portion of Applicant's background art (specification page 2, lines 5-6 and 10-18) fails to teach or suggest the features that are neither taught nor suggested by Bentley. For example, the background art teaches a network management system wherein the network management system is stopped in order to update or add new managed objects in a containment tree. Consequently, the asserted combination fails to teach or suggest all of the claimed features.

Moreover, in the Office Action dated January 29, 2003, the Patent Office admits that Bentley fails to teach or suggest "that the adding and modifying of managed object occurs without interrupting an operation of the network management system." See p. 3, paragraph 4. Applicant's Background Art also fails to teach or suggest these features, and the Patent Office does not appear to rely upon Applicant's Background Art to teach or suggest this feature. Moreover, there is no reference asserted against claim 4 that teaches or suggests this feature. Consequently, for this additional reason, the asserted combination of references fails to teach or suggest all of the claimed features.

For at least above discussed reasons, the asserted combination of references fails to teach or suggest all of the claimed features, as required by Section 103. Because a prima facie case of obviousness cannot be made, it is respectfully submitted that the rejection is improper and should be withdrawn.

### 3. Issue 3: Claim 12

Claim 12 stands rejected under 35 U.S.C. §103(a) over Bentley, in view of Applicants Background Art, and further in view of Sheard et al. (U.S. Patent No. 6,208,345) (hereinafter Sheard).

The asserted combination of references fails to establish a prima facie case of obviousness, as required by Section 103. For example, Claim 12 depends from claim 8. As discussed above, Bentley fails to teach or suggest all the features of claim 8. Moreover, neither

the recited portion of Applicant's background art nor Sheard teaches or suggests the features that are neither taught nor suggested by Bentley.

Applicant's Background Art is discussed above. That discussion is incorporated herein by reference.

Moreover, Sheard relates to a visual data integration system architecture and method. In particular, Sheard discloses a single user interface that provides information to users. Sheard also discloses a processing thread may make use of additional system resources. See column 15, lines 47-50.

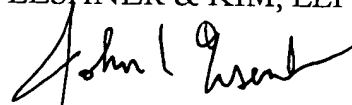
Applicant's Background Art and Sheard both fail to disclose creating a dedicated agent that takes charge of subsequent management operations from the management system requesting an association if an additional thread can be created. Moreover, the Patent Office does not appear to rely on these secondary references to teach such features. Consequently, it is respectfully submitted that a prima facie case of obviousness cannot be made.

For at least above discussed reasons, the asserted combination of references fails to teach or suggest all of the claimed features, as required by Section 103. Because a prima facie case of obviousness cannot be made, it is respectfully submitted that the rejection is improper and should be withdrawn.

**CONCLUSION**

For all of the above reasons, Applicants respectfully request this honorable Board to reverse the rejections of claims 1-15, 17, and 18.

Respectfully submitted,  
FLESHNER & KIM, LLP



Daniel Y.J. Kim  
Registration No. 36, 186  
John C. Eisenhart  
Registration No. 38,128

P.O. Box 221200  
Chantilly, Virginia 20153-1200  
703 502-9440 DYK/JCE:cre:jlg  
**Date: November 12, 2003**

**Please direct all correspondence to Customer Number 34610**

APPENDIX

1. A network management system, comprising:

a management system kernel that provides management systems with a run-time environment; and

a managed object generation environment that provides a development environment for managing application, wherein the management system kernel can at least one of dynamically add and dynamically modify managed object (MO) information based upon an external meta file (EMM) from the managed object generation environment without interrupting an operation of the network management system.

2. The system of claim 1, wherein the management system kernel comprises:

a communication module that provides communication with a network manager;

a managed object framework that maintains information on MO classes;

a kernel that stores a dynamic class loading module and initializes the network management system, wherein said kernel establishes an association with other management systems through the communication module, performs management operations on MOs, adds the MO information in the managed object framework using the dynamic class loading module and modifies the MO information in the managed object framework using the dynamic class loading module; and



a containment tree that organizes MO instances according to the information on MO classes and allows access to the MO instances when a management operation is performed in the network managed system.

3. The system of claim 2, wherein managed object framework maintains information on MO classes by registering MO class codes on a class information table.

4. The system of claim 2, wherein the kernel creates at least one dedicated agent to perform subsequent management operation from management systems with which an association has been established.

5. The system of claim 1, wherein the managed object generation environment comprises:

a MO compiler that complies a MO script to generate the EMM file and MO class codes; and

a dynamic library storing the MO class codes.

6. The system of claim 5, wherein the EMM file includes MO class definition described in the MO script, and identifies a location and name in the dynamic library of a corresponding MO class.



7. The system of claim 5, wherein the MO class codes are compiled and stored in the dynamic library in a form of a dynamic link library.

8. A network management method comprising:

- (a) storing a dynamic class loading routine in a management system kernel;
- (b) initializing a managed system by constructing a managed object framework of the management system kernel that contains information of managed object (MO) classes;
- (c) creating MO instances and registering the MO instances in a containment tree of the management system kernel according to the information of MO classes;
- (d) checking whether a dynamic class loading flag is on when receiving a management operation request from a management system; and
- (e) updating MO information on the management system kernel without interrupting an operation of the management system by,
  - waiting for all threads to complete execution,
  - loading a dynamic library to the managed object framework utilizing the dynamic class loading routine when the dynamic class loading flag is on, and
  - resetting the dynamic class loading flag to off.

9. The method of claim 8, comprising:
  - (f) performing the requested management operation and sending a management operation result to a management system requesting the management operation when the dynamic class loading flag is not on.
10. The method of claim 8, wherein dynamic class loading routine of (e) comprises:
  - opening a EMM file stored outside the management system kernel; and
  - loading the dynamic library indicated by the EMM file.
11. The method of claim 10, further comprising storing information about the management system requesting a management operation.
12. The method of claim 10, further comprising:
  - checking whether an additional thread can be created;
  - creating a dedicated agent to take charge of subsequent management operations from the management system requesting an association if an additional thread can be created;
  - and
  - executing the dedicated agent thread and delivering association and management operation information to the dedicated agent to be utilized in interacting with the management system.

13. The method of claim 8, wherein the dynamic class loading flag is set on when the management system requesting a management operation invokes the dynamic class loading function to perform one of adding and modifying MO information in the management system kernel.

14. The method of claim 13, wherein the management system requesting the management operation invokes the dynamic class loading function by sending a control signal.

15. A network management method, comprising:

- storing a dynamic class loading routine in a management system kernel of the managed system;
- updating the management system kernel by modifying managed object (MO) information in the management system kernel while the managed system is operating by utilizing the dynamic class loading routine; and
- generating the MO information to be modified and generating a external meta file (EMM) in a managed object generation environment of the managed system wherein the dynamic class loading routine opens the EMM file to modify the MO information in the management system kernel..

16. Cancelled

17. The method of claim 16, wherein the MO information to be modified is stored in the managed object generation environment in the form of a dynamic link library.

18. A method of claim 17, wherein the EMM indicates an address of a dynamic link library corresponding to the MO information to be modified, and wherein the MO information is modified in the management system kernel according to said address.